Portofolio 2

, Morten Gade, Anders Hjuldmand, Sigrid Bom, Gustav Helms

2/10/2020

```
#Setup
library(pacman)
pacman::p_load("tidyverse", "lme4", "jpeg")

#Loading the data
df <- read.csv("sleepstudy.csv", header = T)
df$Subject <- as.factor(df$Subject)
```

Task 1: Linear regression

1.a: Make a constant vector of the same length as the data, consisting of ones.

```
# Making a subset for participant 372
Reaction372 <- subset(df$Reaction, df$Subject == 372)
Days372 <- subset(df$Days, df$Subject == 372)

#Making the constant vector
Constant <- rep(1,10)
```

1.b: Report the inner product (aka dot product) of the days vector and the constant vector.

```
# Finding the inner product
sum(Constant*Days372)

## [1] 45
```

1.c: What does the dot product say about the possibility of finding an optimal linear regression?

If the inner product is 0 then the two vectors are orthogonal meaning they are uncorrelated. Since the dot product of the two vectors are 45, they are correlated to some extend. We can conclude that it is possible to find an optimal linear regression of the two vectors.

1.d: Create a 10x2 matrix called X with the days vector and constant vector as columns and use the least squares method manually to find the optimal coefficients (i.e. slope and intercept) to reaction time.

```
#Making a matrix with days and constant
dayMatrix <- matrix(c(Days372, Constant), ncol = 2)
```

```
beta<-solve(t(dayMatrix)%*%dayMatrix)%*%t(dayMatrix)%*%Reaction372
beta

##             [,1]
## [1,]  11.29807
## [2,] 267.04480
```

The slope is 11.3, and the intercept is 267

1.e: Check result using lm(). Use the formula lm(Reaction372~0+X) - the zero removes the default constant.

```
summary(lm(Reaction372~ 0+dayMatrix))

##
## Call:
## lm(formula = Reaction372 ~ 0 + dayMatrix)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -25.064  -4.181   1.008   7.485  11.712
##
## Coefficients:
##            Estimate Std. Error t value Pr(>|t|)
## dayMatrix1   11.298      1.242   9.094 1.72e-05 ***
## dayMatrix2  267.045      6.632  40.265 1.59e-10 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 11.28 on 8 degrees of freedom
## Multiple R-squared:  0.999,  Adjusted R-squared:  0.9988
## F-statistic:  4010 on 2 and 8 DF,  p-value: 9.866e-13
```

The result is the same.

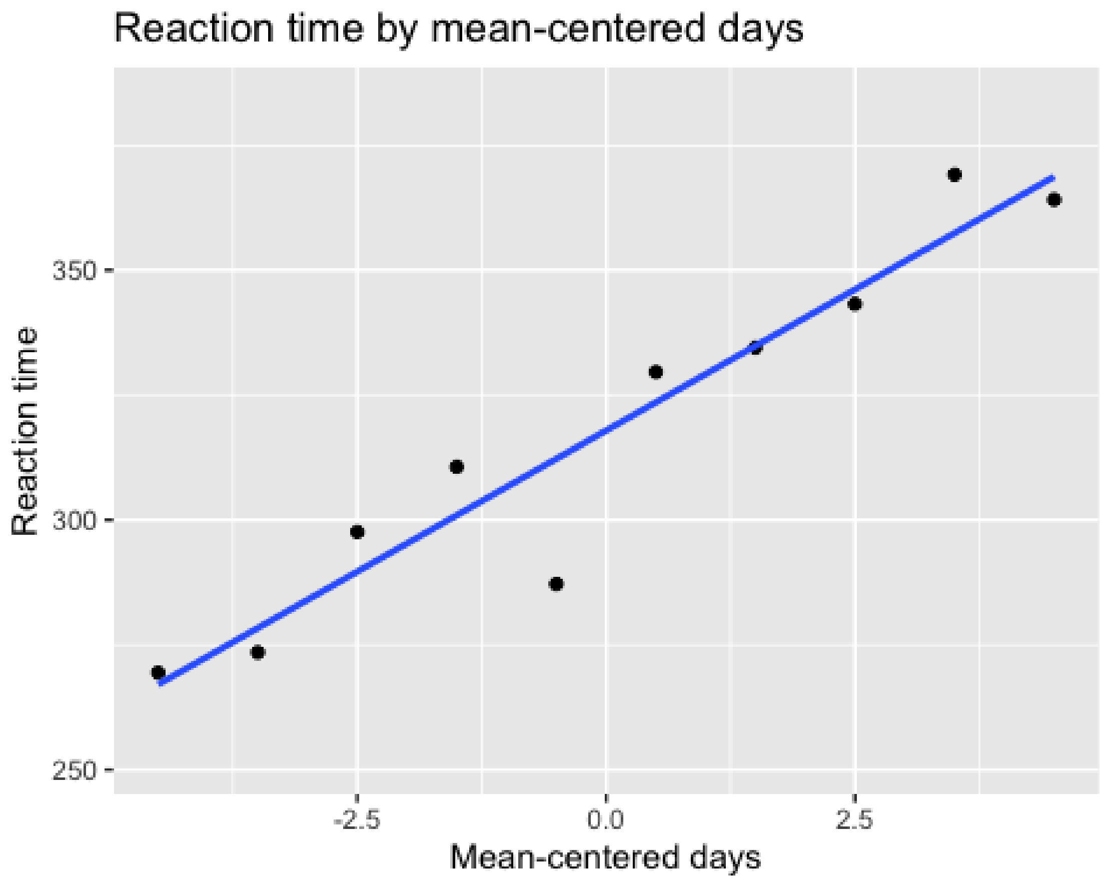
1.f: Subtract the mean of Days372 from the Days372 vector. Replace the days vector with the new vector in X and redo the linnear regression. Did the coefficients change? (we will return to why this happened in a later class, but if you are curious, you can check this website out: https://www.theanalysisfactor.com/center-on-the-mean/)

```
MinusdayMatrix <- (Days372-mean(Days372))
dayMatrix <- matrix(c(MinusdayMatrix, Constant), ncol = 2)

beta<-solve(t(dayMatrix)%*%dayMatrix)%*%t(dayMatrix)%*%Reaction372
beta

##             [,1]
## [1,]  11.29807
## [2,] 317.88613
```

The intercept has changed but the slope is still the same.

1.g: Make a scatter plot with the mean-centered days covariate against response time and add the best fitted line

```
##ggplot
subject372 <- subset(df, Subject == 372)
subject372$Days <- subject372$Days-(sum(subject372$Days)/10)

ggplot(subject372, aes(subject372$Days, subject372$Reaction))+
  geom_point()+
  geom_smooth(method = lm, alpha = 0)+
  labs(x= "Mean-centered days", y="Reaction time", title = "Reaction time by mean-
centered days" )
```



Task 2: Images and matrices

```
#Importing the picture
matrix<-readJPEG('broca.jpg', native = FALSE)
```

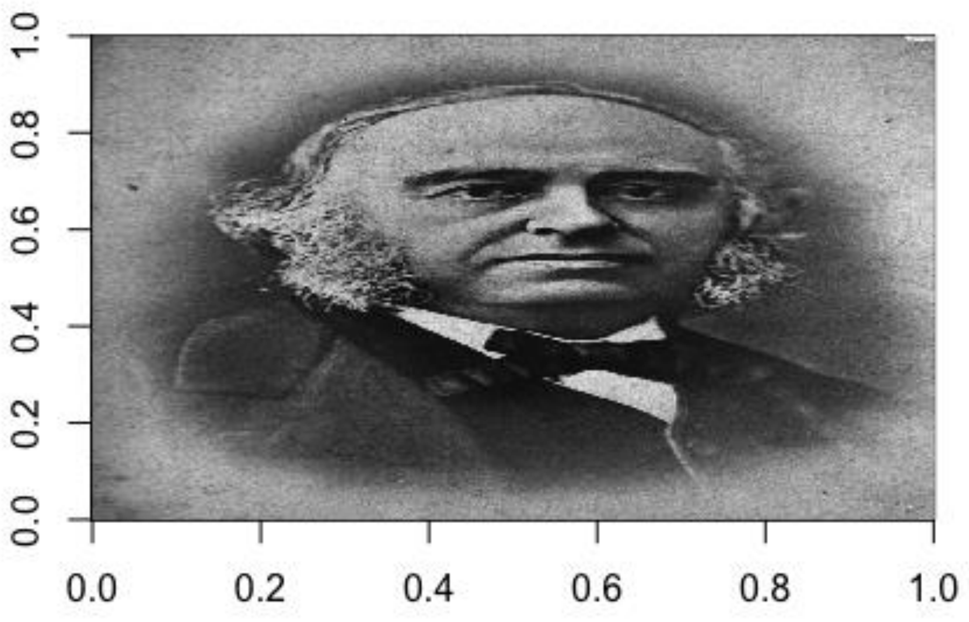
2.a: report how many rows and how many columns the matrix has. What are the maximun, minimum and mean pixel values?

```r
#Number of rows
nrow(matrix)

## [1] 900

# 900 rows

#Number of collumns
ncol(matrix)

## [1] 606

# 606 collumns

#Maximum value
max(matrix)

## [1] 1

# 1 is the highest value

#Minimum value
min(matrix)

## [1] 0.0627451

# 0.0627451 is the minimum value

#Mean value
mean(matrix)

## [1] 0.5118474

# 0.5118474 is the mean value
```

2.b: Make an image of the loaded matrix. Be sure to rotate the image into the correct orientation. The functions needed are found the in lecture slides. Furthermore, grey scale the picture with gray(1:100/100) - this will color values near 0 black/dark and values near 1 white/light.

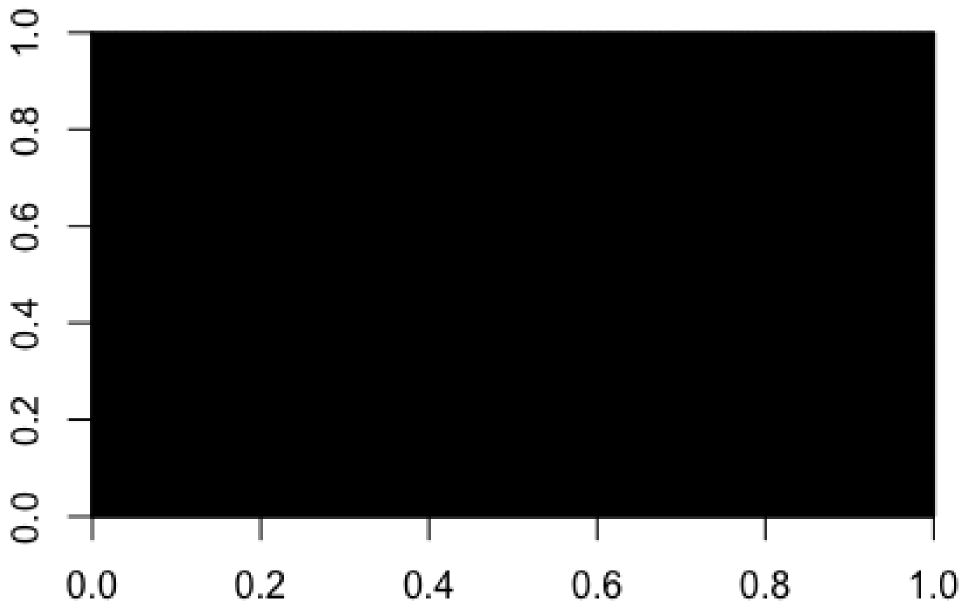
```r
# making a function to rotate the picture
rotate <- function(x) t(apply(x, 2, rev))

#Rotating the picture
broca <- rotate(matrix)
# OBS: The picture now have 606 rows and 900 collumns
```

```r
#convert to image
image(broca, col=gray(1:100/100))
```



2.c: Draw an image with the same dimensions as that from 2.b. But this image should be completely black (hint: use zeros).

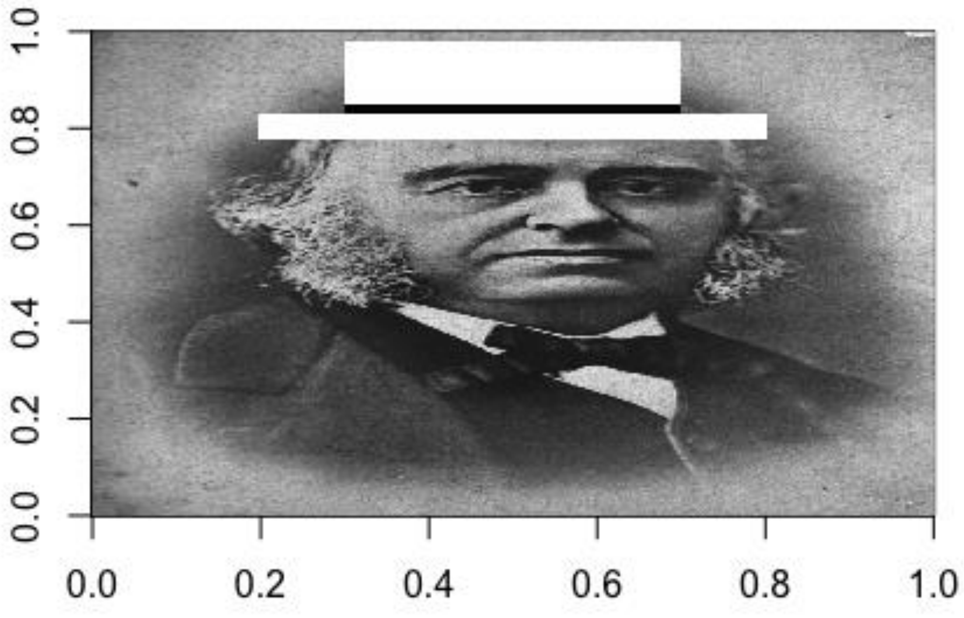
```r
#Making a complete black image
black <- matrix(c(rep(0)), nrow = 606, ncol = 900)

#Drawing the image
image(black, col = grey(seq(0,1)))
```



2.d: Draw a white hat on the image from 2.b (hint: use ones).

```
#Reset
broca <- rotate(matrix)

#making the long flat line
broca[round(0.2*606):round(0.8*606), #Collumns
      round(0.78*900):round(0.83*900)] <- 1 #Rows

#Making the high hat
broca[round(0.3*606):round(0.7*606), #Collumns
      round(0.83*900):round(0.98*900)] <- 1 #Rows

#Making a stripe
broca[round(0.3*606):round(0.7*606), #Collumns
      round(0.83*900):round(0.85*900)] <- 0.01 #Rows


#REMEBER: 606 rows, 900 collumns or [606,900] = [R,C] - Apparently does not count
[C,R] = [900,606]
```

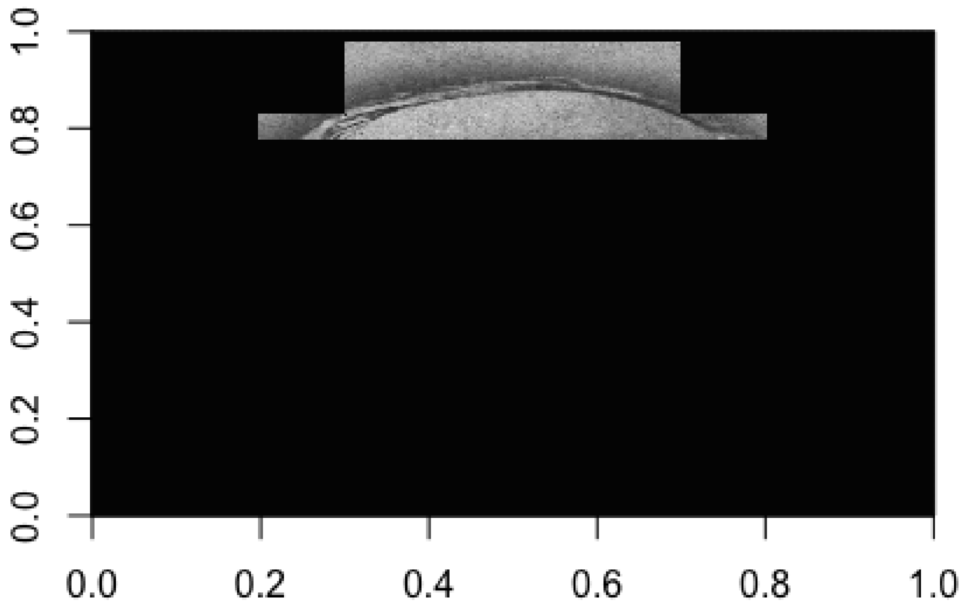
```r
#Drawing the picture
image(broca, col = gray(1:100/100))
```



2.e: Make an image which has the same dimensions as 2.b., and which only contains the parts which was hidden behind the hat in 2.d. The rest should be black.

```r
#Reset
broca1 <- rotate(matrix)

#making the long flat line
black[round(0.2*606):round(0.8*606),
      round(0.78*900):round(0.83*900)] <-
  broca1[round(0.2*606):round(0.8*606),
      round(0.78*900):round(0.83*900)]

#Making the high hat
black[round(0.3*606):round(0.7*606),
      round(0.83*900):round(0.98*900)] <-
  broca1[round(0.3*606):round(0.7*606),
      round(0.83*900):round(0.98*900)]
```

```
#Drawing the picture
image(black, col = gray(1:100/100))
```

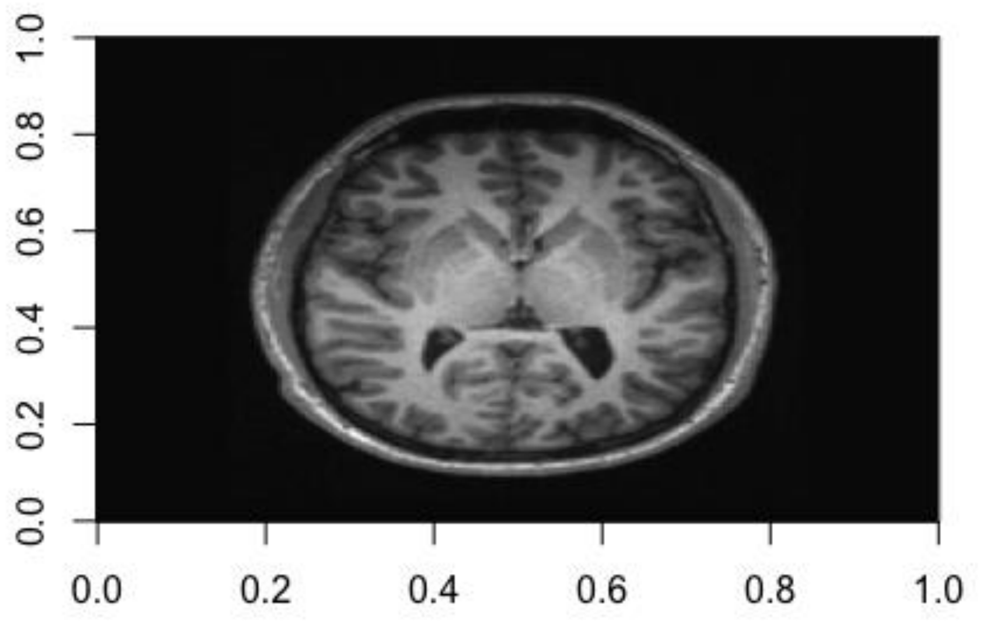


Task 3: Brain and matrices

3.a: Make an image of the brain.

```
#Importing the brain image
brain <- readJPEG("hjerne.jpg", native = FALSE)

#Rotating
brain <- rotate(brain)

#Number of rows
nrow(brain)

## [1] 512

# 512 rows

#Number of collumns
ncol(brain)
```

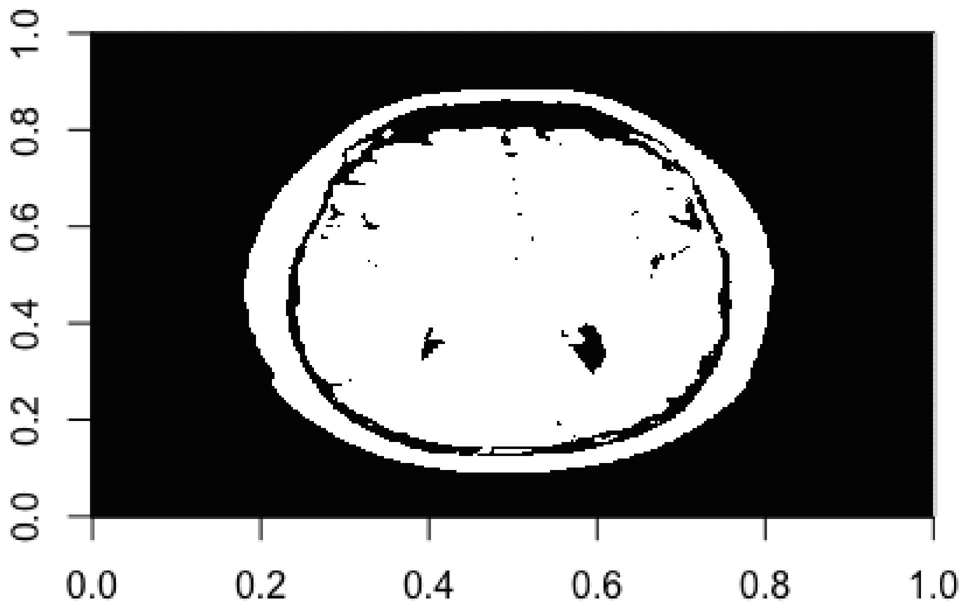
```
## [1] 512

# 512 collumns


#Image
image(brain, col=gray(1:100/100))
```



3.b: We will attempt to find the interesting areas of this brain image, e.g. only areas with gray matter. To do this we will create two masks, one that filters all darker areas away, and one that filters the white matter away. The masks will work by having zeros at the areas we want to fliter away, and ones at the interesting areas. Thus, the mask will have the intended effect if we do element-wise mutiplication of it with the brain matrix. Start by making an image which is white (have ones) where the pixel values of the brain image are larger than the mean value of the whole image. Let the image be black (have zeros) everywhere else. Call this matrix mask1.

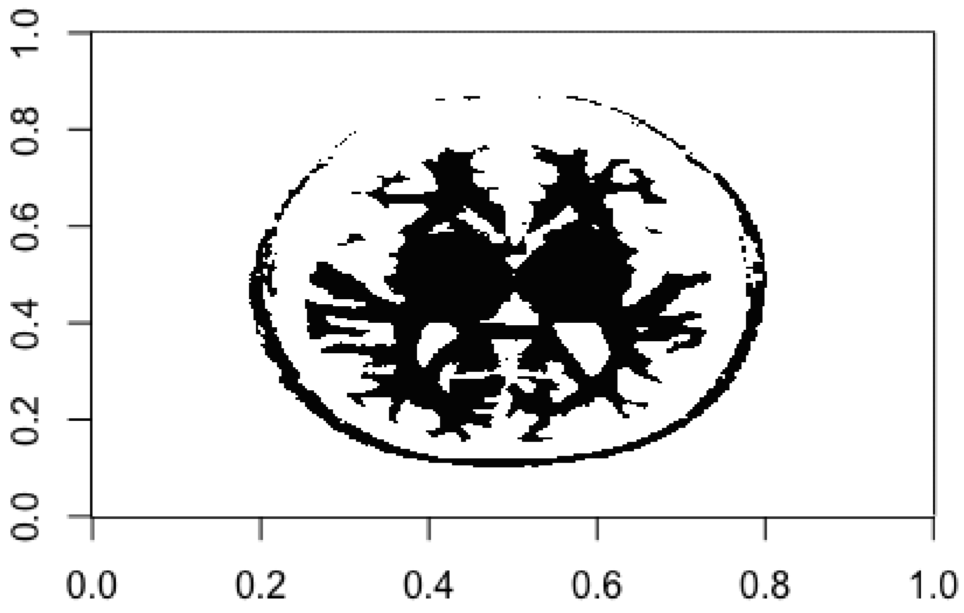
```
#Making the image white when the pixel values of the brain image are larger than
the mean value of the whole image
mask1 <- ifelse(brain > mean(brain), 1, 0)
```

```r
image(mask1, col=gray(1:100/100))
```



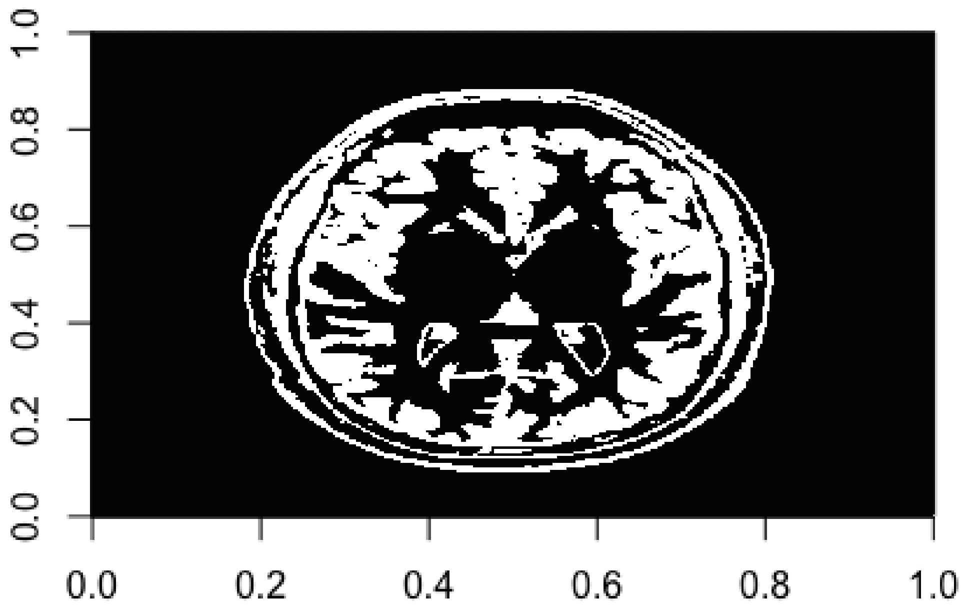
3.c: Make an image which is white where the pixel values of the brain image are smaller than 2.5 times the mean value of the whole image. Call this matrix mask2

```r
mask2 <- ifelse(brain < 2.5*mean(brain), 1, 0)


image(mask2, col=gray(1:100/100))
```



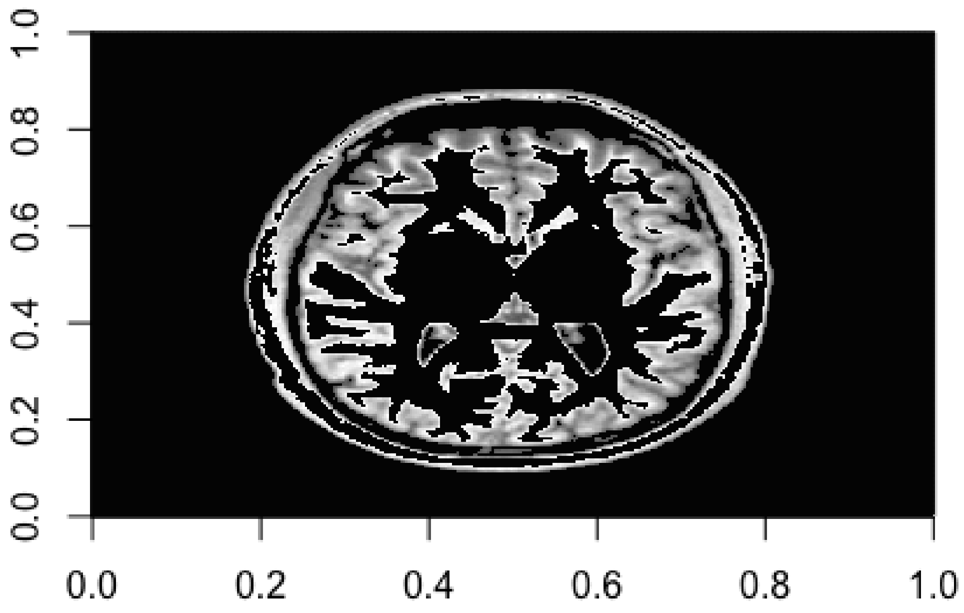
3.d: Convert mask1 and mask2 into one mask with ones where the two masks overlap and zeros everywhere else. What type mathematical procedure can be used to produce this?

```
mask <- ifelse(mask1 == 1 & mask2 ==1, 1,0)
#or
# mask3 <- ifelse(brain > mean(brain) & brain < 2.5*mean(brain), 1,0)

image(mask, col=gray(1:100/100))
```



3.e. Use the combined mask on the brain image to give you an image with only the image values where the mask has ones, and zeros everywhere else. Did we successfully limit our image to only contain gray matter?

```
brain1 <- ifelse(mask == 1, brain, 0)

image(brain1, col=gray(1:100/100))
```



3.e: Count the number of pixels in the combined mask.

```
#Pixels only white
length(mask[mask == 1])

## [1] 50004

# 50.004

#Total number of pixels
length(mask)

## [1] 262144

# 262.144
```

Task 4: Two equations with two unkwons

4.a: In the friday bar, men were three times as likely as women to buy beer. A total of 116 beers were sold. Women were twice as likely as men to buy wine. 92 glasses of wine were sold. How many men and women attended the Friday bar?

```
#Two equations
# 3x1+x2=116
# x1+2x2=92
#x1 = males and x2 = females

# A*X=B <=> AI*A*X = B*AI <=> I*X=B*AI <=> X=B*AI
a <- matrix(c(3,1,1,2), nrow = 2, byrow = T)
b <- c(116,92)
x <- b%*%solve(a)
x

##      [,1] [,2]
## [1,]   28   32
```

There are 28 males and 32 females in the friday bar.